MASTER'S THESIS

Detecting different types of workarounds in IT systems with clustering algorithms

Spoel van der, P (Patrick)

Award date:
2020

Detecting different types of workarounds in IT systems with clustering algorithms

Degree program:	Open University of the Netherlands, Faculty of Management, Science & Technology
	Business Process Management & IT master's program
Course:	IM0602 BPMIT Graduation Assignment Preparation
	IM9806 Business Process Management and IT Graduation Assignment
Student:	Patrick van der Spoel
Identification number:	
Date:	Juni 2020
Thesis supervisor:	Dr. Lloyd Rutledge
Second reader:	Dr. Guy Janssens
Version number:	1.0
Status:	Final version

Abstract

This research shows how and to what extent data clustering algorithms can be used to detect different types of workarounds in structured data of IT systems. Finding them can reduce the risk of lacking data quality and offers opportunities for system improvements. Existing workaround studies place more emphasis on process mining and techniques such as observations, questionnaires and interviews to find them. The Resilience Mining Masters Circle provides a complete research for detecting workarounds in IT systems with data mining techniques. This study focuses in particular on detecting different types of workarounds with clustering algorithms using the CRISP-DM approach. Three experiments have been set up within an artificial data set to detect expressions of the workaround types fictious entity, overcome inadequate IT functionality and misuse of a (text) field with the clustering algorithms k-Means (fast), k-Medoids, Agglomerative Clustering and DBSCAN. The experiments show that multiple clustering algorithms are suitable for detecting the workaround types fictious entity and misuse of a (text) field. The type of overcome inadequate IT functionality was not found in this study. Because this was an artificial data set, it is advisable to repeat this research on a real-life data set and with other expressions of the workaround types examined to solidify this conclusion.

Key words: Workarounds, IT systems, clustering, algorithms, data mining, CRISP-DM, k-Means(fast), DBSCAN, k-Medoids, Agglomerative Clustering

Table of contents

1. Introduction

Shadow IT, Feral Practices and workarounds are different terms in common that they all relate to the use or design of processes and systems without the knowledge and approval of an IT department (Silic & Back, 2014). Research shows that the terms are similar and yet different. According to Kopper & Westner (2016), Feral Practices is the umbrella concept that covers all Shadow IT expressions. Shadow IT is unofficial IT that is added to existing IT, where workarounds are about misusing existing official IT (Kopper & Westner, 2016). A database, which is the source of this research, is an official IT system and therefore is a workaround the most appropriate term that we will use from now on. Workarounds are often applied by users when a system fails to support and meet the wishes of users (Kopper & Westner, 2016). It can help employees and other individuals to work around the limitations of existing organizational processes or systems (Alter, 2014).

Strong (2004) identifies data leaks and loss, compliance issues and undermining official systems as risks for using workarounds. It also ensures data analytics and decision-making becomes more difficult due to the decreased data quality (Drum, Pernsteiner, & Revak, 2017). Besides these risks, workarounds can be very efficient and effective when used in place of the formal systems (Behrens & Sedera, 2004). Gaining insight into the use of workarounds can provide information about the threats, but especially about the opportunities in IT-systems (Silic & Back, 2014).

Workarounds point to shortcomings of current IT systems and are often designated as an opportunity to improve systems permanently (Kopper, 2017). According to Vos (2018), workarounds are performed for a reason. They may contain information about the use of an information system that can be very valuable. Workarounds must be understood and analyzed to improve systems (Vos, 2018). The need to find workarounds within Information Systems efficiently and effectively is enormous (Furstenau, Rothe, Sandner, & Anapliotis, 2016). Detecting workarounds through data mining could contribute to this.

The dataset, which is called 'Betis dataset', is provided by the Open University as educational content. This is not a real dataset of an existing company, but it's a realistic example for a fictious company. It is known that several types of workarounds have been incorporated into the dataset, which have been devised based on experiences and examples in real datasets. Being able to automatically detect these workarounds could lead to system improvements.

This thesis, which is part of an umbrella research by the Resilience Mining Masters Circle, focusses on how and to what extent data clustering algorithms can be used to detect different types workarounds in a database. Other researchers of this circle focus on other data and text mining techniques. The table below shows the contribution each researcher makes to the total research of the Resilience Mining Masters Circle:

	Free text fields	**Structured data**
Clustering	(Spronk, 2020)	This thesis
Prediction	(ten Cate, 2020)	(Huisman, 2020)
Association Rule Mining	(van Rouwendal, 2020)	(Sandfort, 2020)
Outlier detection	(Koskamp, 2020)	

Table 1: Related work of the Resilience Mining Masters Circle

The objective of this research is to detect the use of different types of workarounds in a database with clustering algorithms. The main research question for this thesis is:

"How and to what extent can clustering algorithms detect different types of workarounds in IT systems?"

To answer this main research questions, some sub-questions are formulated. A number of sub-questions are answered as a result of the literature study. These sub-questions form the building blocks for the experiments to be carried out. The other part of the sub-questions is answered with the results from the experiments. The following sub-questions are answered with the results from the literature study:

1. What are indicators of workarounds in data and what type of workarounds are known?
2. What makes data mining a suitable technique for detecting workarounds?
3. Which clustering algorithms are suitable to detect different types of workarounds?
4. How can the performance of the clustering algorithms be measured and compared?

The other sub-questions are answered with the results from the experiments. The following sub-questions have been formulated for this:

5. How and to what extent are clustering algorithms suitable for detecting the workaround type 'fictious entity'?
6. How and to what extent are clustering algorithms suitable for detecting the workaround type ‘overcome inadequate IT functionality’?
7. How and to what extent are clustering algorithms suitable for detecting the workaround type ‘misuse of a (text) field’?

This thesis continues in chapter 2 with the literature review and the corresponding conclusions. The research approach applied in this research is described in Chapter 3. The results of the experiments will be presented in chapter 4 and the conclusion, discussion and recommendations for further research follow in chapter 5.

2. Theoretical Framework

2.1 Research approach

The purpose of this literature study is to gather as much existing knowledge as possible to develop a theoretical framework with which sub-questions can be answered. The answers to these sub-questions form the building blocks for the experiments to be performed, which are of great importance for answering the main question. Literature research has been done on various topics such as workarounds, data mining and (data) clustering algorithms. This literature review answers the following sub questions:

1. What are indicators of workarounds in data and what type of workarounds are known?
2. What makes data mining a suitable technique for detecting workarounds?
3. Which clustering algorithms are suitable to detect different types of workarounds?
4. How can the performance of the clustering algorithms be measured and compared?

2.2 Implementation

The literature study was started via the library of the Open University. Many scientific sources are searched through this library. With the search terms "Shadow IT" and ‘’Workarounds ‘’ a first attempt has been made to find suitable literature. This has led to 16 useful articles that have been used in this thesis. The articles by Alter (2014) and Kopper & Westner (2006) in particular have provided a great deal of insight into the definition and expressions of workarounds.

The following article search took place on ''Data mining'' and ''CRISP-DM''. Six interesting articles were found. There were several relevant articles for data mining, but for a brief overview of what data mining entails, the articles of Mohamada & Tasir (2014) and Chen, Yu & Han (1997) were chosen. The article by Wirth & Hipp (2002) provided good insights for CRISP-DM.

The third search was about data clustering. Eight useful articles have emerged, with the Patel & Thakral (2016) article in particular being very valuable due to the comparison of many clustering techniques. Four interesting sources were found for evaluating the algorithms.

2.3 Results of the literature research

This section presents the results and conclusions from the literature study that has been done with the search terms described in the previous section.

2.3.1 Workarounds

As explained in the Introduction, the term ‘’workaround’’ is defined as the most appropriate term to use in this research. Alter (2014) defined the following definition of a workaround:

‘’A workaround is a goal-driven adaptation, improvisation, or other change to one or more aspects of an existing work system in order to overcome, bypass, or minimize the impact of obstacles, exceptions, anomalies, mishaps, established practices, management expectations, or structural constraints that are perceived as preventing that work system or its participants from achieving a desired level of efficiency, effectiveness, or other organizational or personal goals.”

An organization is mostly not aware that the employees are applying other solutions for completing the job. If an organization is aware of this, it is often unknown to them which workarounds the

employees use. This workaround behavior can have different results for the organization and for the employees (Drum, Pernsteiner, & Revak, 2016). Systems can be found too slow, personal goals conflicts with organizational goals or if an IT-system malfunctions (Alter, 2014).

A workaround is generally considered as a negative phenomenon, but according to the literature there are also positive sides to workarounds. Poelmans (1999) said that end users work around the system to save time and efforts or to avoid the limitations of the system. According to Alter (2014), workarounds are creative solutions that may need to replace an official process that is not functioning properly. Kopper (2017) name workarounds as an opportunity for system improvement. Detecting workarounds in our dataset can lead to system improvements, what makes this research very relevant.

Alter (2014), Patterson (2018) and Outmazgin & Soffer (2013) describe different types of workarounds that they have been able to form from previous studies of workarounds. If we place the workarounds from our dataset next to these types, we can speak of three types of workarounds in the context of our research:

Workaround	Description	Type
Future transport date	The transport date 01-01-2099 occurs regularly and may indicate a cancellation order. However, this is a fictitious date so that we can speak of a fictious entity.	Fictious entity (Outmazgin & Soffer, 2013)
Text or characters in phone number field	Different types of text are used in the phone number field to indicate something, such as missing. There is no unambiguous way how the absence of a telephone number must be represented.	Overcome inadequate IT functionality (Alter, 2014)
Double phone number	Multiple phone numbers are entered in the TEL field that is intended for 1 phone number. There is no other field for an additional phone number and therefore they are both put in the same field.	Overcome inadequate IT functionality (Alter, 2014)
Email address in notation field	There is no field for an e-mail address, so it is entered in a note field that is not intended for this. Text is expected but it is now filled with data, resulting in misuse of fields.	Misuse of (text) fields (Patterson, 2018)

Table 2: Different type of workarounds in the Betis dataset

In this subsection it has become clear what workarounds are, why they are important and what type of workarounds are present in our dataset. This provides an answer to the sub-question: *'What are indicators of workarounds in data and what type of workarounds are known?"*

2.3.2 Data mining

Data mining, also known as the Knowledge Discovery in Database (KDD), is a powerful way to uncover (hidden) information from large volumes of data (Mohamada & Tasir, 2014). It can be used on large datasets that would be difficult to examine sufficiently with traditional approaches. Data mining is a step in the KDD process which consists applying data analysis and discovery algorithms that produce patterns over the data. The discovered knowledge can be applied to improve current systems and decision-making (Chen, Yu, & Han, 1997).

In this research, a data mining experiment is set up using the Cross-Industry Standard Process for Data Mining (CRISP-DM) methodology. According to a survey of Mariscal, Marbán, & Fernánde (2010), the CRISP-DM is the standard for developing data mining projects. This methodology consists six stages which are all organized, structured and defined. A data mining project could be easily revised and understood (Azevedo & Santos, 2008). That makes this methodology suitable for this research.

The data mining reference model consists of the following phases:

1) **Business understanding**: focuses on objectives and requirements from a business perspective.
2) **Data understanding**: initial data collection and proceed with activities to get familiar with the data and to identify data quality problems.
3**) Data preparation**: covers all activities to construct the final dataset from the original data. This contains record, table and attribute selection, data cleaning and transformation of data.
4) **Modeling**: Clustering algorithms are selected and applied. The parameters of this algorithms are calibrated to optimal values.
5) **Evaluation:** Evaluate the model and review the steps that are executed to construct the model.
6) **Deployment**: The knowledge gained from the research must be clearly presented so that it can be used (Wirth & Hipp, 2000).

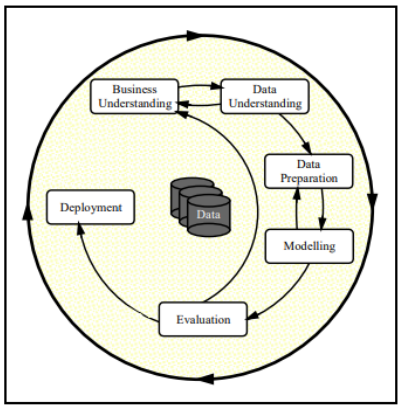

Figure 1: CRISP-DM methodology (Wirth & Hipp, 2000)

The conclusion from this subsection is that data mining is a suitable means to discover patterns in large data sets and that corresponds to the approach of this research to detect workarounds in the Betis data set. It can also be concluded that CRISP-DM is the most complete and the most common methodology for a structured data mining project. This answers the sub-question: *"What makes data mining a suitable technique for detecting workarounds?"*

2.3.3 Data clustering

Jain, Murty & Flinn (2009) describing clustering analyses as ''the organization of a collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity''. It is also known as the unsupervised classification of patterns into groups. Intuitively, patterns within a valid cluster are more similar to each other than they are to a pattern belonging to a different cluster (Jain, Murty, & Flynn, 2009).

Cluster methods are traditionally unsupervised, which means that nothing is known about possible relationships between observations in the dataset and there is no outcome measure. Cluster methods that can be applied to (partially) labeled data are called semi-supervised clustering methods. These methods use both labeled and non-labeled data (Bair, 2013). Because the workarounds in this dataset are known and are labeled, this study involves semi-supervised clustering.

In this study, due to the large number of existing clustering algorithms, a choice was made as to which algorithms are used in the experiments. A selection has been made of popular and commonly used algorithms, which for that reason form a relevant starting point. This is partly based on multiple studies (Chormunge & Jena, 2015) (Bijl, 2018) (Rodriguez, et al., 2016) (Patel & Thakral, 2018) in which different clustering algorithms are applied and compared. When choosing algorithms, the

most important thing is that they are applicable to and suitable for the characteristics of the dataset. It is also important that these algorithms are applicable in RapidMiner, the tool in which the experiments will take place. The following algorithms are therefore used:

K-Means (fast)
''K-means clustering technique is the most effective algorithm in clustering analysis that is applied on data'' (Patel & Thakral, 2018)

The K-means algorithm is a partition-based algorithm and is one of the most common cluster algorithms because of its convergence speed and its simplicity of implementation. It produces relatively high-quality clusters considering the low level of computation that is required. (Dalton, Ballarin, & Brun, 2009). We use the k-means (fast) because of its speed and effectiveness.

K-Medoids
"K-medoids is an improvement of K-Means to deal with discrete data, which takes the data point, most near the center of datapoints, as the representative of the corresponding cluster" (Xu & Tian, 2015)

In k-Medoids clustering, representative objects (medoids) are considered instead of centroids. This method is based on the most centrally located object in a cluster. The difference with the k-Means algorithm is that the k-Medoids are less sensitive to outliers. It is therefore interesting in the context of this research what this algorithm can offer.

DBSCAN
''The well-known clustering technique is DBSCAN which broadly used in applications where mandatory to recognize outlier or to distinguish clusters having arbitrary sizes'' (Patel & Thakral, 2018).

The DBSCAN is a density-based clustering algorithm which finds several clusters that starts from the (estimated) density distribution of corresponding nodes. This algorithm can discover clusters with arbitrary size and shapes. It regards clusters as dense regions of objects that are separated by regions of low-density objects (Verma, Renu, & Gaur, 2014). A characteristic of this algorithm is that it is very suitable for recognizing outliers.

Agglomerative (hierarchical) clustering
''Hierarchical clustering and k-means clustering are arguably the two most popular algorithms used due to their simplicity in result interpretation'' (Tanaseichuk, et al., 2015).

Agglomerative clustering works with a set of individual data points and creates a cluster by merging the two most similar points. The two most similar clusters, which can also be individual data points, are merged with each step. This continues until all data points have been merged into one cluster (Bair, 2013). Agglomerative clustering is a suitable algorithm for this experiment because of its simplicity in interpreting results, just like with the k-Means algorithm.

The conclusion from this subsection is that the k-Means, k-Medoids, Agglomerative Clustering and DBSCAN algorithms are suitable for application to our dataset. This answers the sub-question: *''Which clustering algorithms are suitable to detect workarounds?''*

2.3.4 Evaluation of the algorithms

A confusion matrix is made based on the results of the experiments. This matrix contains information about predicted and actual classifications that are done by a classification system. Performance of such systems is evaluated using the data in the matrix (Kohavi & Provost, 1998). This method is normally used for classification, but because the workarounds are known and therefore it is a semi-supervised study, this is a suitable method to evaluate the results.

		Predicted class	
		yes	no
Actual class	yes	true positive	false negative
	no	false positive	true negative

Figure 2: Confusion matrix (Witten & Frank, 2005)

There are two correct classifications in a confusion matrix, namely the true positives (TP) and the true negatives (TN). With true positives, both the prediction and the outcome are positive, while with true negatives, the prediction and the outcome are both negative. When the outcome is incorrectly predicted as positive, while it is actually negative, we speak of false positives (FP). False negatives (FN) are the exact opposite and stands for incorrectly predicting a negative outcome when it is actually positive (Witten & Frank, 2005).

Some useful evaluation scores can be calculated from this matrix. Because we expect an imbalance in the dataset for the workaround classes, accuracy can be misleading and is therefore not a suitable measure (Wallace & Dahabreh, 2014) We therefore calculate the precision, recall and F-measure. Precision is about the predictive power of the algorithm and estimate the predictive value of a label. The recall score determines the effectiveness of the algorithm on a single class and shows the probability that a positive label is true (Sokolova, Japkowicz, & Szpakowicz, 2006). The F-measure is designed to balance between precision and recall (Witten & Frank, 2005). Because precision and recall have an inverse relationship (high-low, low-high), we focus on the F-Measure. This performance measure is considered to be very effective with unbalanced data (Dal Pozzolo, Caelen, Johnson, & Bontempi, 2015).

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

$$\text{precision} = \frac{TP}{TP+FP}$$

$$\text{recall} = \frac{TP}{TP+FN}$$

$$\text{F} - \text{measure} = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}}$$

Figure 3: Formulas of evaluation scores (Witten & Frank, 2005)

This subsection answers the sub-question: *"How can the performance of clustering algorithms be measured and compared?"*

2.4 Objective of the follow-up research

Revealing that the look of most studies was to investigate how workaround behavior can affect the daily activities of organizations and how to take preventive or corrective measures. The focus was on why and where workarounds were being used. It is important to look at how the workarounds can be detected. Previous workaround studies took place with other techniques. Blijleven, Koelemeijer and Jaspers (2017) used interviews, observations and qualitative coding techniques. Outmazgin (2013) mainly used interviews and questionnaires. There is no comparable research done on detecting workarounds with data mining techniques. The only articles that comes close to this research topic was about process mining in event logs (Outmazgin & Soffer, 2013) and about detecting workarounds with process mining techniques (Vos, 2018). This research attempts to detect different

types workarounds in a database using data mining techniques. Because data mining is very suitable for finding patterns in large volumes of data, this research might be very relevant and valuable.

2.5 Conclusions of the literature research

The information that emerged from the literature study and with which the sub-questions were answered, forms the basis for the continuation of this study. Much knowledge has already been gathered about workarounds in previous studies. This literature study taught us what workarounds are, what type of workarounds there are, why data mining is suitable for finding workarounds, which clustering algorithms are can be used and how to evaluate them. The experiments will attempt to detect three different types of workarounds with the k-Means, k-Medoids, Agglomerative Clustering and DBSCAN algorithms. These three experiments will provide answers to answer the other sub-questions, with which the research question can ultimately be answered.

3. Methodology

This research has a quantitative experimental approach with the aim of answering the research question: *"How and to what extent can clustering algorithms detect different types of workarounds in IT systems?".* Three different experiments are performed with the k-Means, k-Medoids, Agglomerative clustering and DBSCAN algorithms to answer the remaining sub questions.

The purpose of this research is to determine by means of experiments whether it is possible to detect different type of workarounds in a fictional database with the use of four clustering algorithms. This chapter consists of the research method (3.1) and a reflection in the field of validity, reliability and ethics (3.2).

3.1 Research method

The focus is on the statistical meaning of the findings from the experiments, so this research has a quantitative experimental approach. The experiments are objective and specific variables, expressed in statistics, are investigated in the form of workarounds (Apuke, 2017). It aims to apply existing clustering algorithms and look how they perform while detecting the workarounds.

As mentioned in chapter 1 and 2, the CRISP-DM methodology is used in this research.

Business understanding
In this phase is the focus on the understanding of the project objectives and requirements from a business perspective and translate this into a data mining problem. The main objective is to detect the different type of workarounds with clustering algorithms. These workarounds are an indication of misalignment between business and IT. The artificial database contains known workarounds and we try to detect it automatically. Finding workarounds in IT systems using clustering algorithms can lead to system improvements. The requirements and project plan are already discussed in the first two chapters of this thesis.

Data understanding
The data set is loaded into RapidMiner and is being explored to become familiar with the data. This consists of several tables, of which only the '*order*' and '*customer'* tables are relevant for this study. These two tables contain workarounds and the other tables contain mostly master data, which makes them out of scope. We look at the type of data, missing values, size of the data set, balance and other striking issues. Especially missing values (clustering algorithms cannot deal with this) and type of data (not every clustering algorithm can handle any kind of data) are relevant for this research. If a workaround contains an extreme value, these may already be visible in the data understanding phase. An extreme value such as the transport date of 01-01-2099 is immediately visible. After observing and understanding the data, we translate this into preparation actions.

Data preparation
The data set must be prepared in this phase for use in the experiments. This preparation consists of further cleaning of the data, selecting features, converting to numerical data and defining truth labels. Because the research is about workarounds in data fields, the text fields are extracted when selecting features. Both tables were missing with multiple features. Clustering algorithms cannot handle missing values, so that they must be replaced, removed or imputed. Because the workarounds are already known, these are added to the dataset as truth label attributes. These labels consist of a Boolean value if the workaround is present. Because the order set is very large, a representative sample of 10% is taken. This sample is stored and will be used in experiment 1. The

customer set is of an acceptable size, which is why nothing is adjusted for the other two experiments.

Modeling
The main objective is to look for clusters that indicate the presence of a workaround. For this there will be four different types of clustering algorithms applied in three different experiments to see how they perform and if they can detect the known workarounds of the dataset. Of all available algorithms in RapidMiner, a choice has been made to use the k-Means, k-Medoids, Agglomerative clustering and DBSCAN. These can handle the type of data from both tables.

A separate model is built for each algorithm. The preliminary work has already been done in the preparation phase; this involves performing the algorithm on the prepared sample. Per experiment and algorithm, we look at what it takes to get a well-functioning model. With the order sample it is important to select the correct attributes, normalize the set before the different clustering algorithms can run. In the customer set, a target role must also be designated per experiment because there are multiple labels. Different parameters will also be applied per algorithm to see what gives the best results. Although the modeling process is quite generic, the setup can change per experiment or algorithm. This is then explained in the relevant experiment.

Evaluation
After performing the experiments, the results of applying the different algorithms will have to be compared with each other. Truth labels have been defined and therefore the performance of the models is evaluated by using a confusion matrix, from which evaluation scores are obtained. Important here are the True Positives, which indicate whether the workarounds are correctly predicted. The False Positives are also important because they tell more about fields that are incorrectly classified as workaround. The precision, recall and F-measure scores are calculated on the basis of the confusion matrix, where we mainly focus on the F-measure because we are dealing with unbalanced workaround labels.

For each algorithm, we will look at which parameters give the best results in evaluation scores. For each experiment, it is examined which algorithm gives the best results to detect a certain type of workaround. At the end, the results of each algorithm for all three experiments are compared.

Deployment
The deployment phase in this research consists of the delivery of this thesis report, in which the results and conclusions of the experiments are incorporated. The results from this study can be used for further research into this subject and for system improvements that emerged from the research.

3.2 Reflection on validity, reliability and ethics

Validity
With internal validity it is about whether the conclusion drawn regarding the investigated relationship is correct (Heala & Twycross, 2015). In this study there are no external influences or variables that may impact the research. All selected algorithms are based on literature research, which is linked to the type of data in the dataset. This benefits the internal validity of the research. Labeling workarounds makes it possible to determine whether all workarounds have been correctly identified, which also increases internal validity.

The external validity is limited because the dataset is not a real, but a realistic presentation of a database for a fictional company. It is generated by an algorithm and the workarounds are also fictional, which can make it easier to detect. An existing database will therefore have more noise

than this dataset. It cannot be said with certainty that the results of this study can be generalized to other databases and it is unknown how well the data represents real-life datasets.

Reliability

Reliability is about whether the same results come from the research if it were to be conducted by another researcher at a different time (Heala & Twycross, 2015). The technical aspects of this research are reliable because of the standard clustering algorithms of RapidMiner and by following the common CRISP-DM methodology. This methodology is reliable because it ensures a structured project. In addition, all settings, choices and substantiations are documented, making them easy to replicate. This contributes to the utility and reliability of this work. Using standard and common algorithms in RapidMiner like k-Means, k-Medoids, Agglomerative clustering and DBSCAN also ensures reliability.

Ethics

There are no ethical issues when writing this report. The data set is an artificial generated data set for a fictional company, so there are no privacy risks. This would be different if it were an existing data set from an existing company. Regarding the software used in this research, a legal license is used for the experiments in RapidMiner.

4. Results

The data is explored and described in chapter 4.1. Subsequently, in section 4.2 the data preparation takes place to prepare the data set for the experiments. Section 4.3 discusses the modeling used in the experiments. Subsequently, section 4.4 describes the first experiment with the 'Transport date' workaround. In section 4.5 the second experiment with the ''Phone number' workaround is described, after which in paragraph 4.6 the third and final experiment with the 'Email address' workaround is performed. Finally, section 4.7 contains a comparison of the results of the various experiments.

4.1 Data understanding

During this phase of the CRISP-DM the data was collected and loaded. Insights have been obtained in the data set about the content and quality of the data. The original Betis data set contains seven tables, of which only the *opdracht* (Order) and *klant* (Customer) table contains relevant workarounds in the structured data. The other tables *errormsg* (Error message), *gebruiker* (User), *locatie* (Location) and *tarief* (Rate) concern master data or in the case of rate a calculation and are therefore not included.

The data from the tables have the Dutch language. Because this research is written in English, a column with the English description has been added in both tables. These terms will in the future be used as names for the attributes. Tables 3 and 4 have been translated directly from RapidMiner after importing the dataset. Because circle members from the Resilience Mining Thesis Circle also conduct research on the same data set, the representation of table 3 and table 4 corresponds to tables from their research (Huisman, 2020) (Koskamp, 2020).

Attribute	Description	Attribute type	Missing values	Min value	Max value
NR	Order number	Integer	0	1623	1778104
KLANTNR	Customer number	Integer	0	242	11513
STRAAT	Street	Polynomial	0	Least: a (1)	Most: Kathalijne U. Kloppertuin (14331)
PC	Postal code	Polynomial	0	Least: 9999XM (1)	Most: 1135PW (3447)
HUISNR	House number	Polynomial	0	Least: b (1)	Most: 2 (25368)
PLAATS	City	Polynomial	0	Least: Zwinderen (1)	Most: Amstelveen (317995)
DOPDR	Order date	Date	0	Mar 15, 1988	Jul 16.210
DPLAN	Planned date	Date	588427	Mar 15, 1988	Jul 14, 2010
DTRANS	Transport date	Date	2671	Mar 15, 1988	Jan 1, 2099
COLLI	Bales	Integer	16515	1	7
KG	Weight	Real	49884	1	5
MDW	Employee	Polynomial	0	Least: ddd (1)	Most: are (103247)
BONBIN	Ticket received	Polynomial	0	Least: J (569194)	Most: N (1070865)
BEDRAG	Amount	Real	0	0	70
NOTITIE	Note	Polynomial	1		

Table 3: Statistics of Order example set in RapidMiner

This order table contains 1,640,059 examples and 15 regular attributes. It contains different types of attributes, namely in numeric fields (integer and real), text fields (polynomial) and date fields (date).

It is striking in these statistics that several fields have missing values. These missing values may have arisen because they are optional fields. They are worth mentioning because clustering algorithms cannot handle missing values. These missing values must therefore be imputed, deleted or replaced in the data preparation process. A large number of values are missing in the *planned date* (588427), *weight* (49884) and *bales* (16515) attributes. In the *transport date* attribute, which contains the workaround, 2671 values are missing. This could indicate that the date of 01-01-2099 is used explicitly as a fictional date.

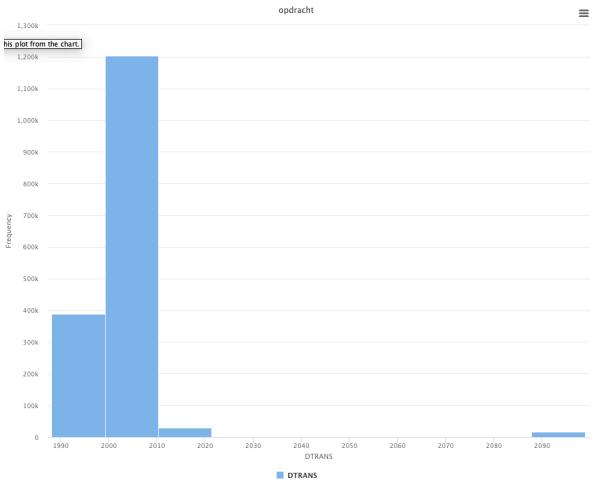

Figure 4: Histogram of the DTRANS (transport date) attribute

If we look at the histogram of the transport date attribute in Figure 4, the extreme value is immediately noticeable on the right. However, the purpose of this research is to be able to trace this type of workaround through clustering algorithms.

Attribute	Description	Attribute type	Missing values	Min value	Max value
NAAM	Name	Polynomial	0	Von lieb […] adding (1)	Andwent Veevoeder (2)
NR	Customer number	Integer	0	242	11515
STRAAT	Street	Polynomial	0	b (1)	Kathalijne U. Kloppertuin (81)
HUISNR	House number	Polynomial	0	b (1)	2 (174)
PC	Zip code	Polynomial	0	9971 BB (1)	1145PT (29)
PLAATS	City	Polynomial	0	Zwijndrecht (1)	Amstelveen (2060)
CP	Contact person	Polynomial	2172	Mw. W.J. Punselie (1)	A. Romney (2)
NOTITIE	Note	Polynomial	4101	zyronst@euronet.nl (1)	Onbekend (5)
TEL	Telephone number	Polynomial	1974	0979839293 (1)	Geen (253)
BLOK	Blocking code	Polynomial	0	J: 62	N: 10370

Table 4: Statistics of Klant (Customer) example set in RapidMiner

The customer table has 10 attributes and consists of 10,432 examples, with which it contains considerably less data than the order table. The customer table contains only two attribute types. The customer number is an integer, the other attributes are text fields (polynomial). Although we investigate data fields in this study, fields like *telephone number* and *note* do contain data that we can investigate.

Also, in this table there are three attributes with missing values, namely the *contact person, telephone number* and *note*. The last two are especially interesting because they also contain workarounds. Various methods are used in the *note* field. Many examples within this attribute are filled with an e-mail address, while there are also 4101 missing values. This says that little use is

made of real notes, but this field is misused with an e-mail address because there is no separate attribute.

4.2 Data preparation

In this phase the data is prepared so that it can be used for the experiments. This preprocessing takes place RapidMiner. Both the order and the customer table are prepared separately by various actions.

Order table

The date values from the *transport date*, *order date* and *planned date* attributes have been converted to numerical values containing the number of days since epoch. A new attribute is generated with the condition that the transport date is 47116 (number of days since epoch). This attribute is called WORKAROUND1_DATE and contains a Boolean value that indicates whether the line contains the workaround. It is also selected as the truth label and therefore the workaround can be validated on the basis of this truth set. The workaround is processed in 16258 examples, which concerns 0.99% of all examples in the order table.

The text attributes *street*, *city*, *postal code*, *employee* and *note* are filtered out because we focus on data fields. Other researchers from the Master Circle focus on text mining (van Rouwendal, 2020) (ten Cate, 2020) (Sandfort, 2020). The order table is very large, which is not necessary to carry out the experiments. It was decided to use a sample of 10% to both have enough data and save time during execution. The order table sample is stored, and the experiments are performed with this sample. It consists of 164,005 examples, of which 1676 examples contain the workaround (1.01%). This implies an imbalance in the generated workaround attribute.

The clustering algorithms cannot handle missing values and it is decided that the missing values should be imputed. Replacing is not an option because the data set then shifts too much. Deleting examples would cause a large loss of data in valuable features.

Figure 5 shows this complete data preparation process in RapidMiner for the Order set:

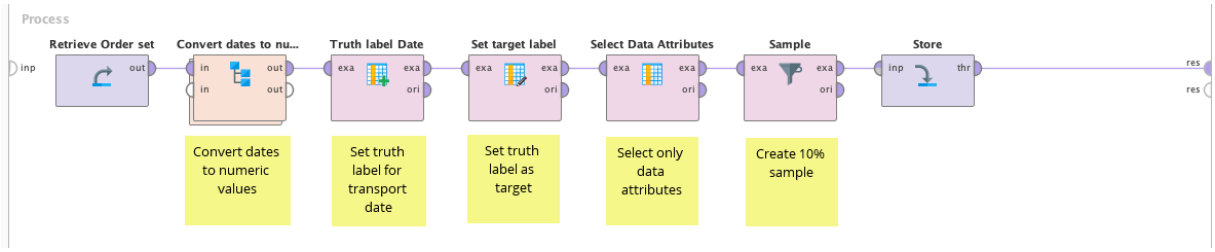

Figure 5: Data preparation process in RapidMiner for the order table

Customer table

First, the two test lines with dummy values are filtered from the table. The customer table contains more workarounds and therefore the attributes WORKAROUND2_PHONE and WORKAROUND3_EMAIL is generated. The condition for WORKAROUND2_PHONE is when the *phone number* field contains text such as "*geen", "onbekend", "-", "bgg", "b.g.g."* or when the length of an example in the *phone number* attribute is longer than 15 characters, it will be marked. This occurs 1126 times, which means that this workaround appears in 10.8% of the fields in the *phone number* attribute. This class is therefore not in balance. The condition for WORKAROUND3_EMAIL consists of all *note* fields that contain the character ''@". This occurs in 6304 examples (60.4%) of the note fields, making this workaround very present and there is therefore no imbalance as with the other two workarounds. The missing values for these generated attributes are set to ''false''.

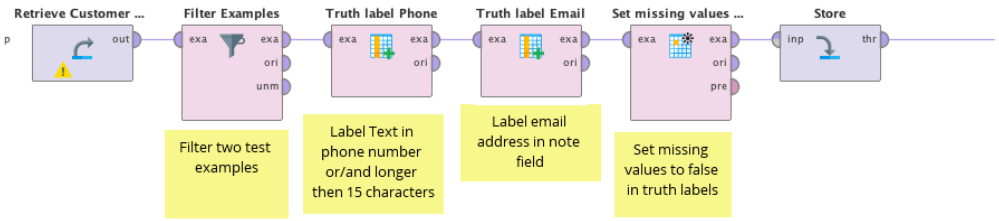

Figure 6: Data preparation process in RapidMiner for the customer table

The missing values in the attributes are also imputed in this table, which was done in a separate process. Replacing these values would change the dataset too much and deleting them would result in a loss of 60% of the examples. Imputation maintains a representative data set. The customer table is a lot smaller than the order table, so the complete table is used instead of a sample.

4.3 Modelling

The data sets are now prepared and stored and form the starting point for the modeling. A generic process is modeled, which is used as much as possible in every experiment. Where this differs from the generic process, such as adding or disabling operators, this will be explained in the concerning experiment. The difference between the two processes is the use of a different data set and the target label has yet to be selected in the customer set via the set role operator.

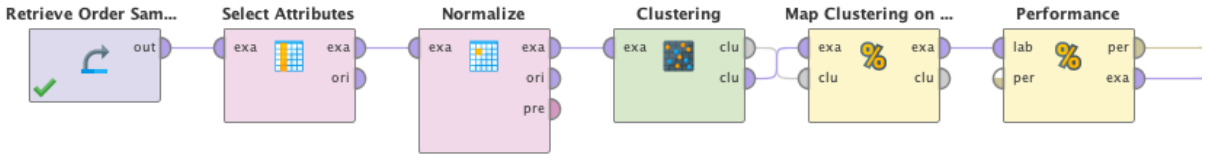

Figure 7: Generic modelling process in Rapid Miner for the order set

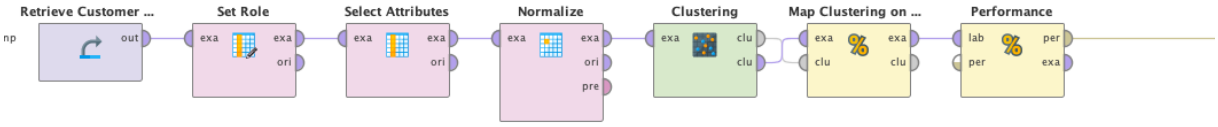

Figure 8: Generic modelling process in Rapid Miner for the customer set

Set Role:

Two labels have been created in the customer set, so that the role of the target label must be adjusted per experiment via the set role operator. In the order set this is already set during the preparation process.

Select features:

Relevant features are selected and experiments can be conducted with them to see what yields the best results.

Normalize:

An important point in clustering is the application of normalization to the data set. If this does not happen, the different scales of the features have too much influence on the distance calculation. The data sets are normalized with the Z-transformation method. It mainly affects numerical values.

Map Clusters on Label:

The Map Clustering on Labels creates prediction labels based on the clusters. The number of clusters

needs to be the same as the number of labels. Because there is only 1 label per experiment, only 2 clusters can be created in each experiment.

Performance:
Because the previous operator creates a prediction label, the performance can be measured with the Performance operator, building a confusion matrix and calculating accuracy, precision, recall and F1.

4.4 Results of experiment 1: Transport date

In this first experiment it is attempted to detect WORKAROUND1_DATE with the transport date of '01-01-2099', or day 47116 after conversion, in the Order set with four different clustering algorithms. Each algorithm uses the numerical measure Euclidean Distance, which calculates the geometric distance between two data points. For this, the Polynomial attributes house number and ticket received have been removed, so that only numerical data remains. As mentioned before, the workaround is present in 1.01% of the dataset and therefore there is class imbalance.

4.4.1 k-Means(fast)

K-Means is a partition-based algorithm which produces relatively high-quality clusters considering the low level of computation that is required. As explained earlier in this chapter, we use k = 2. With that we got the following results:

f_measure: 100.00% (positive class: true)

	true false	true true	class precision
pred. false	162329	0	100.00%
pred. true	0	1676	100.00%
class recall	100.00%	100.00%	

Figure 9: Confusion matrix of the K-Means(fast) algorithm in experiment 1

Figure 9 shows that k-Means(fast) works very well with a precision, recall and F-measure of 100%. With such scores, there is a probability of overfitting. It is also possible that the workaround stands out in such a way (perhaps because of the artificial nature of the dataset) and is therefore easy to find.

4.4.2 k-Medoids

This is a similar process to k-Means. This is a fairly slow and burdensome algorithm, which forces us to take a sample. A sample of 50%, 30% and 10% was taken. The results of all three samples are very poor. The 10% sample even gave a result with 0% precision, 0% recall and 0% f-measure. The 30% sample with k=2 yielded the following result:

f_measure: 5.84% (positive class: true)

	true false	true true	class precision
pred. false	33047	0	100.00%
pred. true	15668	486	3.01%
class recall	67.84%	100.00%	

Figure 10: Confusion matrix of the k-Medoids algorithm in experiment 1

The f-measure is very low with a score of only 5.84%. A large number of false positives are incorrectly predicted, so the precision score is also low. k-Medoids has difficulty detecting the fictitious transport date, even though it is very different from the other data. We can conclude that the k-Medoids algorithm is unable to find the workaround label.

4.4.3 Agglomerative clustering

The Agglomerative clustering process looks slightly different than the process of the other algorithms. In addition to using a different algorithm, the 'Flatten Clustering' operator has also been added between the Agglomerative clustering operator and the Map Clustering on Labels operator. The Agglomerative clustering operator provide a cluster hierarchy, where these clusters merge at a certain distance. Based on the given hierarchical cluster model, Flatten Clustering creates a flat cluster model by extending nodes in the order of their distance. This continues until two clusters are created.

As a Mode in the Agglomerative Clustering parameters, the choice was initially for Complete link because it is sensitive to outliers. Due to the different fictitious transport date, we think that this will achieve the best result. These parameters gave the following results:

f_measure: 100.00% (positive class: true)

	true false	true true	class precision
pred. false	162329	0	100.00%
pred. true	0	1676	100.00%
class recall	100.00%	100.00%	

Figure 11: Confusion matrix of Agglomerative Clustering in experiment 1

Changing the mode from Complete to Average or Single link did not change the result. All three modes yielded the same perfect results with only True Positives and True negatives and 100% scores. We therefore suspect that this is overfitting.

4.4.4 DBSCAN

The DBSCAN algorithm is applied, which is a density-based algorithm. Compared to the generic process, only the algorithm has been adapted to DBSCAN. DBSCAN is not about the value in K, but in Epsilon, which specifies the size of the neighborhood. The number of minimum points is also given, which is a minimum number of points that forms a cluster. Because epsilon and min points are variable, the optimize parameters (grid) operator is used to calculate the best values.

DBSCAN is also a very heavy and time-consuming algorithm, especially if you want to calculate the optimal parameters. A 10% sample (16,400 examples) was used to perform the algorithm. The optimizing parameters operator is set with an epsilon from 0.0 to 2.5 and the minimum points from 0 to 25. Figure 12 shows the results:

DBSCAN.epsilon	DBSCAN.min_points	precision	recall	f_measure ↓
1.500	20	0.873	1	0.932
1.400	18	0.863	1	0.927
1.400	19	0.863	1	0.927
1.400	16	0.863	1	0.927
1.400	20	0.863	1	0.927
1.400	17	0.863	1	0.927
1.300	18	0.827	1	0.905
1.300	15	0.827	1	0.905
1.300	19	0.827	1	0.905
1.300	16	0.827	1	0.905
1.300	20	0.827	1	0.905
1.300	17	0.827	1	0.905
1.200	13	0.806	1	0.893
1.200	15	0.802	1	0.890
1.200	18	0.802	1	0.890
1.200	19	0.802	1	0.890

Figure 12: Optimized parameters DBSCAN in experiment 1

From this we can deduce that there are many variables that score very high. It is striking that the recall is always 100% and that the precision increases as the epsilon increases. However, higher than 1.5 did not give better results. It is also noticeable that the precision for a particular epsilon remains the same despite the variation in min points. The size of the neighborhood is therefore more relevant than the minimum number of points that form a cluster. An epsilon of 1.5 with 20 minimum points gives the best results with an f-measure score of 93.22%:

f_measure: 93.22% (positive class: true)

	true false	true true	class precision
pred. false	16219	0	100.00%
pred. true	23	158	87.29%
class recall	99.86%	100.00%	

Figure 13: Confusion matrix of DBSCAN in experiment 1

4.3.5 Conclusion experiment 1: Transport date

All four algorithms have been applied and each application has resulted in a confusion matrix. These are compared below to determine whether one or more algorithms are able to detect workaround 1 with the fictious transport date.

Experiment 1: Transport date	**Precision**	**Recall**	**F-1**
k-Means(fast)	100%	100%	100%
k-Medoids	3.01%	100%	5.84%
Agglomerative Clustering	100%	100%	100%
DBSCAN	87.29%	100%	93.22%

Table 5: results of experiment 1: Transport date

Despite the imbalance in the label class, the k-Means(fast) and Agglomerative Clustering have a perfect score of 100%. It is therefore suspected that this is an overfitting. DBSCAN also scores very well in this experiment with an f-measure of 93.22% and a high precision score. The k-Medoids algorithm gave very poor results, which may be due to imbalance in the label class. This experiment shows that three algorithms are able to detect the workaround type fictious entity, which answers sub-question 5: *How and to what extent are clustering algorithms suitable for detecting the workaround type 'fictious entity'?*

4.5 Results of experiment 2: Phone number

This second experiment attempts to detect WORKAROUND2_PHONE in the customer set. This dataset contains two labeled attributes, so we still have to set the WORKAROUND2_PHONE attribute as the target here using the set role operator. Almost all attributes are nominal values, except the *customer number*.

attribute	weight
WORKAROUND2_PHONE	0.000
NR	0.000
BLOK	0.001
PLAATS	0.009
HUISNR	0.110
STRAAT	0.447
PC	0.693
TEL	0.716
CP	0.784
NOTITIE	0.955
NAAM	0.959

Figure 14: Weight by Information Gain for Workaround2_Phone

Figure 14 shows that NR (*customer number*) should have no relevance for this workaround, so the decision was initially made to remove *customer number* and use the measure type Nominal Measures. The results turned out better with Mixed Euclidean Distance (for nominal and numerical values) than with Nominal Measures and therefore we use this measure type in this experiment with all attributes. There is class imbalance, where the workaround is present in only 10.8% of the examples.

4.5.1 k-Means (fast)

k-Means (fast) shows poor results in this experiment:

f_measure: 18.00% (positive class: true)

	true false	true true	class precision
pred. false	4655	555	89.35%
pred. true	4649	571	10.94%
class recall	50.03%	50.71%	

Figure 15: Confusion matrix of the K-Means(fast) algorithm in experiment 2

The f-measure is very low at 18%. There are many false positives, at the expense of the precision score. The number of false negatives is also considerable, so that the recall is not too high.

4.5.2 k-Medoids

The k-Medoids model runs with k = 2 and only has an f-measure of 21.22%. This scores slightly better than k-Means (fast), but the result is still poor.

f_measure: 21.22% (positive class: true)

	true false	true true	class precision
pred. false	4709	447	91.33%
pred. true	4595	679	12.87%
class recall	50.61%	60.30%	

Figure 16: Confusion matrix of the k-Medoids algorithm in experiment 2

Many false negatives and false positives mean that the algorithm cannot predict well, as expressed in the low percentages for precision and the f-measure. As with k-Means (fast), there are too many false negatives and false positives in the result.

4.5.3 Agglomerative Clustering

In this model, selecting Complete Link as mode with the Agglomerative clustering algorithm ensures the best result:

f_measure: 17.65% (positive class: true)

	true false	true true	class precision
pred. false	4739	575	89.18%
pred. true	4565	551	10.77%
class recall	50.94%	48.93%	

Figure 17: Confusion matrix of the Agglomerative Clustering algorithm in experiment 2

This algorithm also performs poorly with this workaround label. Switching from mode to Average or Single produced even worse results. As with the previous two algorithms, where the scores do not differ much, there are too many false positives and false negatives. This results in low recall, precision and f-measure.

4.5.4 DBSCAN

The parameters are first optimized to read the best scores for epsilon and minimum points. This was done with epsilon 1 to 3 and minimum points 0 to 100, which led to the following results:

DBSCAN.epsilon	DBSCAN.min_points	precision	recall	f_measure ↓
2.500	70	0.164	0.575	0.255
2.500	80	0.160	0.615	0.254
2.500	60	0.164	0.546	0.252
2.500	90	0.156	0.631	0.250
2.600	90	0.166	0.495	0.248
2.600	80	0.167	0.486	0.248
2.600	100	0.165	0.504	0.248
2.500	50	0.161	0.520	0.246
2.500	100	0.152	0.641	0.245
2.600	70	0.160	0.453	0.236
2.500	40	0.148	0.464	0.225
2.600	60	0.145	0.398	0.213
2.500	30	0.138	0.409	0.207
2.600	50	0.142	0.375	0.206

Figure 18: Optimized parameter results for DBSCAN in experiment 2

Both an epsilon of 2.5 and 2.6 are among the highest results. With parameters from 0.0 to 3.0 epsilon, it is striking that 2.5 and 2.6 give the best results. Nevertheless, the f-measure is low and fluctuates between 20.6 and 25.5%. The results are best with an epsilon of 2.5 and minimum points of 70:

f_measure: 25.46% (positive class: true)

	true false	true true	class precision
pred. false	5994	479	92.60%
pred. true	3310	647	16.35%
class recall	64.42%	57.46%	

Figure 19: Confusion matrix of the DBSCAN algorithm in experiment 2

Many false positives and relatively many false negatives, which means that DBSCAN cannot make a correct prediction in this experiment. DBSCAN scores comparable results with the other three algorithms, although DBSCAN has the highest f-measure score. Relatively, this algorithm scored fewer false positives and false negatives, which means that the numbers for true positives and true negatives are slightly higher than with the other algorithms.

4.5.5 Conclusion experiment 2: Phone number

The table below shows the results from experiment 2, which attempted to detect text in phone number fields and multiple phone numbers within 1 field via WORKAROUND2_PHONE.

Experiment 2: Phone number	Precision	Recall	F-1
k-Means (fast)	10.94%	50.71%	18.00%
k-Medoids	12.87%	60.30%	21.22%
Agglomerative Clustering	10.77%	48.93%	17.65%
DBSCAN	16.35%	57.45%	25.46%

Table 6: Results of experiment 2: Phone number

The results of this experiment are very poor. No algorithm is able to detect these two manifestations of workarounds to overcome inadequate IT functionality. A possible reason for this is that there was a large class imbalance in the label that may or may not contain the workaround. It can be concluded that in the case of class imbalance these four algorithms are unable to detect this type of workaround. This therefore provides a clear answer to sub-question 6: *How and to what extent are clustering algorithms suitable for detecting the workaround type 'overcome inadequate IT functionality'?*

4.6 Results of experiment 3: Email address

In experiment 3 we searched for WORKAROUND3_EMAIL, which consists of the email address in the note field of the customer set. The workaround occurs in 60.4% of the examples, which makes the class a lot more balanced than the workarounds in the other experiments. The set role operator is now being modified and WORKAROUND3_EMAIL is now selected as the target label.

attribute	weight
WORKAROUND2_PHONE	0.000
NR	0.000
BLOK	0.001
PLAATS	0.009
HUISNR	0.110
STRAAT	0.447
PC	0.693
TEL	0.716
CP	0.784
NOTITIE	0.955
NAAM	0.959

Figure 20: Weight by Information Gain for WORKAROUND3_EMAIL

The weight by information gain operator also indicates here that the *customer number* has no relevance on the label. With the results from experiment 2 in mind, we try both measure types in this experiment. It soon turns out that the Nominal measures usually works better as measure type, which is why the *customer number* is removed and the normalization operator disabled.

4.6.1 k-Means(fast)

The k-Means (fast) model for the third experiment has been performed and gives the following results:

f_measure: 74.22% (positive class: true)

	true false	true true	class precision
pred. false	166	247	40.19%
pred. true	3960	6057	60.47%
class recall	4.02%	96.08%	

Figure 21: Confusion matrix of the k-Means(fast) algorithm in experiment 3

Figure 21 shows that this algorithm scores a nice f-measure of 74.22%. The number of false positives is still quite high. Due to the high number of false positives, the precision is slightly lower than you would hope. The recall of the negative class is therefore very low with 4.02%.

4.6.2 k-Medoids

The k-Medoids algorithm scores well in this experiment an f-measure of 84.53%. Despite the fact that there are only 2 clusters, k-Medoids is able to find the algorithm in a cluster.

f_measure: 84.53% (positive class: true)

	true false	true true	class precision
pred. false	3226	1030	75.80%
pred. true	900	5274	85.42%
class recall	78.19%	83.66%	

Figure 22: Confusion matrix of the k-Medoids algorithm in experiment 3

The number of false positives is considerably lower than with k-Means (fast), which means that the precision is a lot higher. The recall is lower because the number of false negatives is larger. Nevertheless, the f-measure of 84.53% ensures a better result.

4.6.3 Agglomerative clustering

The Agglomerative Clustering model scores best with the Average link mode:

f_measure: 75.37% (positive class: true)

	true false	true true	class precision
pred. false	33	16	67.35%
pred. true	4093	6288	60.57%
class recall	0.80%	99.75%	

Figure 23: Confusion matrix of the Agglomerative Clustering in experiment 3

This algorithm scores high on recall and f-measure. The precision score lags somewhat behind due to the high number of false positives. With an f-measure of 75.37%, this algorithm appears to be able to find the workaround.

4.6.4 DBSCAN

The parameters are first optimized to read the best scores for epsilon and minimum points. This was done with an epsilon between 0.0 and 3.0 and min points 0 to 100, which led to the following results. No results were obtained with nominal measures as measure type. In this case we switched to Mixed Euclidean Distance, all attributes are selected, and normalization is enabled again. We ran the same optimize parameters operator again and came to the following results:

DBSCAN.epsilon	DBSCAN.min_points	precision	recall	f_measure ↓
2.400	40	0.694	0.955	0.804
2.400	50	0.692	0.957	0.803
2.300	30	0.685	0.969	0.803
2.400	60	0.689	0.961	0.803
2.300	40	0.680	0.973	0.800
2.400	70	0.680	0.968	0.799
2.400	30	0.697	0.934	0.798
2.300	20	0.687	0.951	0.798
2.400	80	0.664	0.979	0.792
2.300	50	0.658	0.984	0.789
2.400	90	0.654	0.989	0.787
2.400	100	0.648	0.992	0.784
2.300	60	0.647	0.993	0.784
2.300	70	0.637	0.996	0.777
2.300	80	0.632	0.997	0.774

Figure 24: Optimized parameters DBSCAN in experiment 3

Figure 24 shows that DBSCAN works best with an epsilon of 2.3 and 2.4. Variation in the number of min points causes minimal differences in the results. An epsilon of 2.4 with min points of 40 has the highest scores:

f_measure: 80.36% (positive class: true)

	true false	true true	class precision
pred. false	1471	286	83.72%
pred. true	2655	6018	69.39%
class recall	35.65%	95.46%	

Figure 25: Confusion matrix of the DBSCAN algorithm in experiment 3

With an f-measure of 80.36%, this algorithm appears to be able to find the workaround. Here too, the precision lags somewhat due to the number of false positives, but the score for f-measure is good.

4.5.5 Conclusion experiment 3: Email address

In the table below, the results of the different algorithms within experiment 3 are compared:

Experiment 3: Email address	Precision	Recall	F-1
k-Means (fast)	60.47%	96.08%	74.22%
k-Medoids	85.42%	83.66%	84.53%
Agglomerative Clustering	60.57%	99.75%	75.37%
DBSCAN	69.39%	95.46%	80.36%

Table 7: Results of experiment 3: Email address

This experiment shows that all four algorithms have been able to detect this type of workaround, misuse of a text field. Especially k-Medoids shows a very good result with a lot higher precision score than the other algorithms, which is at the expense of the recall score. Although only 2 clusters could be formed, they could all find the label in a cluster. This experiment answers sub-question 7: *How and to what extent are clustering algorithms suitable for detecting the workaround type 'misuse of a (text) field''?*

4.6 Comparison of the results

The ability to create clusters in a data set to discover temporary solutions is more important than the accuracy of each model when comparing the algorithms. The results of all three experiments with the four different algorithms are shown in Table 8:

Experiment 1: Transport date (workaround type: fictious entity)	**Precision**	**Recall**	**F-1**
k-Means (fast)	100%	100%	100%
k-Medoids	3.01%	100%	5.84%
Agglomerative Clustering	100%	100%	100%
DBSCAN	87.29%	100%	93.22%
Experiment 2: Phone number (workaround type: inadequate IT functionality)	**Precision**	**Recall**	**F-1**
k-Means (fast)	10.94%	50.71%	18.00%
k-Medoids	12.87%	60.30%	21.22%
Agglomerative Clustering	10.77%	48.93%	17.65%
DBSCAN	16.35%	57.45%	25.46%
Experiment 3: Email address (workaround type: misuse of text field)	**Precision**	**Recall**	**F-1**
k-Means (fast)	60.47%	96.08%	74.22%
k-Medoids	85.42%	83.66%	84.53%
Agglomerative Clustering	60.57%	99.75%	75.37%
DBSCAN	69.39%	95.46%	80.36%

Table 8: Results of all experiments

The three different experiments show different results. Experiment 1 shows that three algorithms score high on detecting a fictious entity, such as the transport date of 01-01-2099. It should be noted that the scores of k-Means (fast) and Agglomerative Clustering are perfect on an unbalanced set, which tends to overfitting.

The results of experiment 2 show that these expressions of inadequate IT functionality such as text in telephone field and entering multiple telephone numbers in one field cannot be found with these clustering algorithms. All four score very poorly. The four algorithms scored very well in experiment 3, which shows that all four were able to find this expression of the misuse of a text field, in the form of an email address in the *note* field.

5. Discussion, conclusions and recommendations

On the basis of the findings from this study, we initiate the discussion, draw conclusions and make recommendations for practice and further research.

5.1 Discussion

Previous literature research already showed that existing literature focused mainly on the meaning of workarounds, the expressions of workarounds, the different types of workarounds and the reasons for their use (Alter, 2014) (Patterson, 2018)(Outmazgin & Soffer, 2013). Existing research that focuses on finding workarounds is mainly done with methods such as interviews, observations and questionnaires (van de Weerd, Beerenpoot, Vollers, & Fantinato, 2019) (Blijleven, Koelemeijer, & Jaspers, 2017)(Outmazgin & Soffer, 2013).

Another study by Vos (2018) focuses on detecting workarounds with process mining techniques. Outmazgin and Soffer (2014) searched for workarounds in event logs with process mining but failed to detect all types. Even more recent research focuses on how to deal with the workarounds when they are detected, but not so much about the method of detection (van de Weerd, Beerenpoot, Vollers, & Fantinato, 2019). The Resilience Mining Thesis Circle distinguishes itself by focusing on data mining as a method to detect workarounds. Our research is specifically about using clustering algorithms to detect type of workarounds. This makes our perspective a valuable addition to the existing methods used in previous research.

Based on the literature defining different types of workarounds, the workarounds from our dataset are plotted within one of the prescribed types. The aim is to see whether clustering algorithms are able to detect different types of workarounds within a data set. The experiments show that within our dataset it is possible to automatically detect two different types of workarounds with different clustering algorithms, despite the fact that one of the workaround labels was heavily imbalanced. This research forms a valuable contribution to existing knowledge and methods about detecting workarounds. Since there is no comparable research which specifically focuses on detecting workarounds with data mining, this research is an important starting point for further research.

5.2 Conclusions

All sub-questions have already been answered with the results of the literature study and the results from the experiments, so that a thorough answer can now also be given to the main question: *"How and to what extent can clustering algorithms detect different types of workarounds in IT systems?".* We conclude from the three experiments that two types of workarounds, namely a fictious entity (experiment 1) and misuse of a text field (experiment 3) can be found in this database and under these conditions with multiple clustering algorithms. The workaround type from experiment 2 (overcome inadequate IT functionality) could not be found by any algorithm. However, we must conclude that the results apply to this situation and further research under other circumstances should show whether this conclusion can be drawn more broadly.

5.3 Recommendations for practice

The results show that in a semi supervised scenario it is possible to detect different workaround types in data fields with multiple clustering algorithms. It turns out that if the workaround is known

and you label it, the clustering algorithms are able to detect it. What we learn from this is that if you look so specifically for predefined workarounds, there is no attention for new information. The focus is on finding these known workarounds, but it does not look at similar connections and correlations between attributes that can indicate a workaround. This departed from a normal clustering scenario as a result of the already known workarounds from the course documentation. Certainly, with clustering, which is normally an unsupervised method, the intention is to look for new information in each cluster. It should be kept in mind that in a typical clustering scenario we don't know what we're looking for. It is therefore advisable to work with this dataset without prior knowledge.

Important choice during this research was how to deal with the imbalance in the workaround labels. It was decided to leave them that way and see what the performance is. Practice shows in experiment 3 that a more balanced label (WORKAROUND3_EMAIL) provides better results from all four algorithms over the other experiments. Repeat studies with balanced labels may improve the results from experiment 1 and especially 2.

5.4 Recommendations for further research

Several points emerged from this study that are interesting for follow-up research. It was an artificial data set that was compiled on the basis of experience and real data sets. It is therefore advisable to perform this research on an existing data set to see what the performance is. In addition, three types of workarounds have now been investigated, but more types of workarounds exist. It would therefore be interesting to investigate whether other types of workarounds could also be found. Also, the expressions of different types of workarounds in this artificial data set are not the only expressions that fall within that type of workaround. We therefore recommend repeating this study with other expressions of a type of workaround.

The data set was also unbalanced, which raises the question of what the performance would be like if the workaround label sampled up or down to balance the class. In experiment 1 there were also perfect scores of two algorithms that tend to overfitting. Follow-up research could clarify this. The fact that we are talking about labels already means that it was not a completely unsupervised study as you would expect with clustering. The workarounds were already known and labeled, making it a semi-supervised study. Finally, we would recommend conducting this research completely unsupervised to see if the different types of workarounds can be found without any foreknowledge.

5.5 Reflection

An artificial dataset can make the workarounds easier to find than in a real-life database. This is at the expense of external validity, so the question is whether the results are the same for an existing data set. A fictional date like ’01-01-2099’ was so different from the rest that it might have affected the performance of some algorithms. There is also a suspicion that there is an overfitting.

Another problem we ran into is that clustering is unsupervised, but due to the labeling of the workarounds, this became a semi-supervised study. This entailed the limitation that you deliberately searched for the workaround, while clustering normally does not have a target label. To measure performance with these clustering algorithms, the Map Clustering on Labels operator had to be used every time. The disadvantage of this was that the number of clusters had to be equal to the number of labels, so that each experiment could only be performed with only 2 clusters. Ideally, we would like to investigate at which number of clusters the results are best by varying this number.

A lot of time was spent in the modeling phase. Algorithms like k-Medoids and DBSCAN are very time consuming and hard to run. As a result, in experiment 1 we had to choose a sample within the already created sample in the data preparation process in order to still get valuable results. Because of this, data may be lost that could influence the results. In order to maintain the validity of the experiment, we would therefore prefer to release the algorithms on the same data sample.

Because no previous research has been done into the operation of automatic detection of workarounds in databases using data mining techniques, this research forms a solid basis for further research into discovering different type of workarounds with clustering algorithms.

Acknowledgements

First of all, I want to thank our thesis supervisor Lloyd Rutledge for his advice, feedback and contribution to the classroom meetings that have helped me a lot. In addition, I want to thank the second reader, Guy Janssen, for his clear feedback during this research. I would also like to express a word of thanks to my fellow group members Ruben, Maarten, Janneke, José, Jan and Thomas for their interesting contributions and discussions during the classroom meetings, from which I learned a lot. Special thanks go to my employers, Peter Dau and Francois Laans from QforIT, for the time and resources that they have offered me to facilitate this thesis. Finally, I want to express my gratitude to the home front, which always offered the support and understanding that I needed.

References

Alter, S. (2014). Theory of Workarounds. *Business Analytics and Information Systems*.

Apuke, O. (2017). Quantitative Research Methods: A Synopsis Approach. *Arabian Journal of Business and Management Review (Kuwait Chapter)*.

Azevedo, A., & Santos, M. (2008). KDD, semma and CRISP-DM: A parallel overview. Amsterdam: IADIS European Conference on Data Mining 2008.

Bair, E. (2013). Semi-supervised clustering methods. *Wiley Interdiscip Rev Comput Stat*, 349-361.

Behrens, S. (2009). Shadow systems: the Good, the Bad and the ugly. *Communications of the ACM*, 124-129.

Behrens, S., & Sedera, W. (2004). Why Do Shadow Systems Exist after an ERP Implementation? Lessons from a Case Study. *Association for Information Systems AIS Electronic Library (AISeL)* .

Bijl, A. (2018). *A comparison of clustering algorithms for face clustering.* Groningen.

Blijleven, V., Koelemeijer, K., & Jaspers, M. (2017). Exploring Workarounds Related to Electronic Health Record System Usage: A Study Protocol. *JMIR Res Protocols*.

Chen, M., Yu, P., & Han, J. (1997). Data mining: An overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering*, 866 - 883.

Chormunge, S., & Jena, S. (2015). Efficiency and Effectiveness of Clustering Algorithms for High Dimensional Data. *International Journal of Computer Applications*.

Dal Pozzolo, A., Caelen, O., Johnson, R., & Bontempi, G. (2015). Calibrating Probability with Undersampling for Unbalanced Classification. *IEEE Symposium Series on Computational Intelligence (SSCI).* Cape Town.

Dalton, L., Ballarin, V., & Brun, M. (2009). Clustering Algorithms: On Learning, Validation, Performance, and Applications to Genomics. *Current Genomics*, 430-445.

Drum, D., Pernsteiner, A., & Revak, A. (2016). Walking a mile in their shoes: user workarounds in a SAP environment. *International Journal of Accounting & Information Management*, 185-204.

Drum, D., Pernsteiner, A., & Revak, A. (2017). Workarounds in an SAP environment: impacts on accounting information quality. *Journal of Accounting & Organizational Change, 13*(1), 44-64.

Furstenau, D., Rothe, H., Sandner, M., & Anapliotis, D. (2016). Shadow IT, Risk, and Shifting Power Relations in Organizations. *22nd Americas Conference on Information Systems, At San Diego, USA*.

Heala, R., & Twycross, A. (2015). Validity and reliability in quantitative research. *Evidence-Based Nursing*, 66-67.

Huisman, J. (2020). *Resilience Mining: Detecting Shadow IT in IT Systems with Data Classification.*

Jain, A., Murty, M., & Flynn, P. (2009). Data Clustering: A Review. *ACM Computing Surveys* , 264-323.

Kohavi, R., & Provost, F. (1998). Glossary of Terms; Special Issue on Applications of Machine Learning and the Knowledge Discovery Process. *Machine Learning*(30), 271-274.

Kopper, A. (2017). Perceptions of IT Managers on Shadow IT. *ORGANIZATIONAL TRANSFORMATION & INFORMATION SYSTEMS (SIGORSA)*, pp. 1-10.

Kopper, A., & Westner, M. (2016). Towards a Taxonomy for Shadow IT. Paper presented at the Americas Conference on Information Systems.

Koskamp, M. (2020). *Mining for workarounds in information systems using outlier detection.*

Mariscal, G., Marban, O., & Fernandez, C. (2010). A survey of data mining and knowledge discovery process models and methodologies. *The Knowledge Engineering Review*, 137-166.

Mohamada, S., & Tasir, Z. (2014). Educational data mining: A review. *Procedia - Social and Behavioral Sciences*, 320-324.

Outmazgin, N. (2013). Exploring Workaround Situations in Business Processes. *Lecture Notes in Business Information Processing*, 426-437.

Outmazgin, N., & Soffer, P. (2013). Business Process Workarounds: What Can and Cannot Be Detected by Process Mining. *Enterprise, Business Process and Information Systems Modeling*, 48-62.

Outmazgin, N., & Soffer, P. (2014). A process mining-based analysis of business process work-arounds. *Software & Systems Modeling*, 309-323.

Patel, K., & Thakral, P. (2018). The Best Clustering Algorithms in Data Mining. *International Conference on Communication and Signal Processing,.* India.

Patterson, E. (2018). Workarounds to intended use of health information technology: A narrative review of the human factors engineering literature. *Human Factors: The Journal of Human Factors and Ergonomics Society*, 281-292.

Poelmans, S. (1999). Workarounds and distributed viscosity in a workflow system: a case study. *ACM SIGGROUP Bull*, 11-12.

Rodriguez, M., Comin, C., Casanova, D., Bruno, O., Amancio, D., & da Costa, L. (2016). Clustering algorithms: A comparative approach. *PLoS ONE*.

Sandfort, T. (2020). *Resilience mining: identifying workarounds in IT systems using association rule data mining.*

Silic, M., & Back, A. (2014). Shadow IT – A view from behind the curtain. *Computers & Security*(Volume 45), 274-283.

Sokolova, M., Japkowicz, N., & Szpakowicz, S. (2006). Beyond Accuracy, F-score and ROC: a Family of Discriminant Measures for Performance Evaluation. *Advances in Artificial Intelligence*(4304), 1015-1021.

Spronk, J. (2020). *Mining for workarounds in text fields with clustering algorithms.*

Strong, D., & Volkoff, O. (2004). A roadmap for enterprise system implementation . *Computer*, 22-29.

Tanaseichuk, O., Khodabakshi, A., Petrov, D., Che, J., Jiang, T., Zhou, B., . . . Zhou, Y. (2015). An Efficient Hierarchical Clustering Algorithm for Large Datasets. *Austin Journal of Proteomics, Bioinformatics & Genomics*.

ten Cate, R. (2020). *Detecting Shadow IT in free text fields using text mining and classification.*

van de Weerd, I., Beerenpoot, I., Vollers, P., & Fantinato, M. (2019). WORKAROUNDS IN RETAIL WORK SYSTEMS: PREVENT, REDESIGN, ADOPT OR IGNORE? *Conference: 27th European Conference on Information Systems (ECIS 2019).*

van Rouwendal, J. (2020). *Tekst en association rule mining voor detecteren van workarounds in vrije tekst die gestructureerde data-invoer omzeilen.*

Verma, T., Renu, R., & Gaur, D. (2014). Implementation of DBSCAN Algorithm using Similarity Measure from Rapid Miner. *International Journal of Applied Information Systems*.

Vos, L. (2018). *Analyzing and Understanding workarounds in an IS to improve Business Processes in a Multinational Corporation.* Amsterdam: Vrije Universiteit.

Wallace, B., & Dahabreh, I. (2014). Improving class probability estimates for imbalanced data. *Knowledge and Information Systems volume*, 33-52.

Wirth, R., & Hipp, J. (2000). CRISP-DM: Towards a Standard Process Model for Data Mining. *Journal of Data Warehousing*.

Witten, I., & Frank, E. (2005). Data Mining: Practical Machine Learning Tools and Techniques. In I. Witten, & E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques* (Vol. 2, pp. 156-168). San Francisco: Elsevier.

Xu, D., & Tian, Y. (2015). A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science*, 165-193.